REMARKS

Claims 1-20 now stand in the application, new claims 18-20 having been added. Reconsideration of the application and allowance of all claims are respectfully requested.

All of claims 1-17 are rejected as unpatentable over Yates in view Beck et al. This rejection is respectfully traversed.

The problem sought to be addressed by the present invention is described at pages 2-3 of the present application. A roaming user may have a profile on the home network that entitles the user to certain services, but the roaming network may not provide those services. Even if service logic for providing the communication service is transferred to the roaming network from the home network, a service execution platform may be missing, or in the case of an internet connection the roaming user may have the service performed by a server on the home network and quite far away, sometimes leading to unacceptable delays.

So the goal of the present invention is to provide communication services to a user independent of the communication network to which the user is connected.

According to the present invention, a service server (SSV) transmits a service container (CONT1) containing a service machine (SM1) to a service computer (SSC). The service computer (SSC) executes the service machine (SM1), the service machine (SM1) managing the execution of a personal service for the communication means (TERA, TERB) which are connected to a communication network (NET). The service computer (SSC) provides at least one network lock (NWL) for the first service container (CONT1), the at least one network lock (NWL) offering to the first service container (CONT1) a predefined interface to the

communication network (NET) for the provision of the personal service. The service machine (SM1) executes or applies at least one service component (CP1, CP2, CP3) transmitted to the service computer (SSC) via the first service container (CONT1) or via a second service container (CONT2), whereby the personal service is provided for the communication means (TERA, TERB). (FIG. 1)

Thus, the service computer performs the function of a network-independent service execution platform executing a service container for providing a personal service to the user. Also important is that the service computer has an interface to the local network, and can offer this as a "network lock" to the service container, with the service server sending a service container that is designed to use the network lock available from that service computer.

As also described at the top of page 6 of the specification, there is also provided a "monitor lock" by which the service server monitors conditions at the service computer, and can send other service container to a different service computer for servicing the user if the first service computer becomes overloaded.

Yates uses a software agent that is made up of a selected set of software modules, with execution of the software agent providing a desired service in a configuration determined by the particular configuration (i.e., the particular selection of modules) of the agent. See, lines 57-65 of column 2. So in the context of claim 1, the software agent would be the claimed service machine. The examiner alleges that the modules of Yates are analogous to the claimed service container, and that the code and SIBB's in the module are analogous to the claimed service machine, but this cannot be. None of the modules are capable of providing a service. Yates uses a software agent as the service machine, and the goal of Yates is to have the agent be

reconfigurable, so that modules are used to permit reconfiguration. A module does not provide a service, but is only a part of the software agent that provides the service.

Lines 3-12 of column 4 describe the existence of several different types of modules. SIBB's which do not provide a service themselves but provide supporting operations, adaptor modules which do not provide a service but instead establish appropriate interfaces and protocols, and coordinator modules which control selection and/or modification of software modules for each configuration. None of these modules is capable by itself of providing a service.

This is particularly apparent in the case of SIBB's, in that their very name, i.e., "service-independent building blocks," would preclude them from providing a specific service.

Thus, as illustrated in Fig. 3 of Yates, the software agent that when executed provides the service is the combined result of different types of modules.

To satisfy the limitations of the presently pending claims, e.g., claim 1, there must be a service computer connected to a user by way of a communications network. The examiner equates this with , e.g., a terminal domain 101 in Fig. 1 of Yates. There must then be a service server that provides to the terminal domain 101 a service machine. This is lacking in Yates. The terminal domain may be provided with a set of software elements from which it can select to build its software agent (service machine), but it does not receive the software agent from somewhere else encapsulated in a container. If the terminal domain is not provided with its own set of software modules, it can be given access to those available at another domain (see, e.g., lines 1-5 of column 5). But it is modules it will have access to, and it can select from those

modules to construct its own desired agent ("machine"). It does not received the machine from somewhere else, and certainly not encapsulated in a container.

The examiner relies on Beck to teach the transmission of a service machine contained within a service container. According to Beck, a service is made up of an interface 402, implementation 403 and adaptor 404 as shown in Fig. 4. As described at lines 3-24 of column 6 and illustrated in Fig. 5, when a client device requests usage of a service but the service is not already loaded onto the device, the three components of the service are downloaded to the device.

There are many important distinctions between the present invention and Beck. For example, the present invention involves a service computer obtaining a service machine from a service server and then running the service machine, and then providing the requested service to a communication means over a communication network. Beck is not downloading a service to a service computer but rather to the communication device itself. Further, since Beck is downloading a service to the device itself, there is no need for a network lock to be used by the service computer to communicate the service to the communicating device via a communication network. One could go on to note many other distinctions, but it is sufficient to note that the only relevance of Beck to the present invention is the generic teaching of downloading software to a device to provide a service when the service is requested and the necessary software is not already resident on the device.

While Beck arguably teaches downloading a service "machine" to a device, it does not teach that the machine is transmitted to the device in a service container. Indeed, the device separately downloads each of the three components of the service.

If one of skill in the art were to consider the teachings of the two references, he would see Yates teaching that a terminal domain could retrieve software modules from another domain in order to construct a desired software agent, and he would see Beck teaching that a mobile telephone device can download software needed to perform a service at that device. One logical combination of the teachings of the two reference might be to use both features, i.e., downloading software to the mobile phone when needed and downloading software to the terminal domain when needed. But this would not get Yates any closer to the claimed invention.

Another view by the ordinary artisan may be that Beck teaches downloading software to a device when needed and Yates teaches downloading software to a domain when needed, and Beck does not use SIBB's so does not have the degree of configuration flexibility of Yates. But in either case, the software is downloaded to the device or domain in multiple parts. There is no suggestion in either reference of the transmission of a container containing the software agent or service.

The examiner is of the position that any software module "encapsulates" the executable code within the module. The examiner refers to the discussion of the Board where it concluded that individual building blocks are contained within a software module in Yates. As an aside, it is submitted that the Board is incorrect, that Yates makes it clear that a SIBB is a module, not that a module contains plural SIBB's. This is clear from lines 3-5 of column 4, but this is not the important point. What is important is that an individual module, even one "encapsulating" a SIBB, is not a service machine. A service machine is something that, when executed, provides a service. An SIBB cannot possibly do that, or it could not be "service-independent" as its name requires. It is clear that in Yates the only thing that could correspond to the claimed service

machine is the software agent that only comes about on combining multiple selected modules. Neither reference suggests encapsulating such a machine and sending it to a service computer for execution.

Since neither of the cited references teaches the encapsulation of a machine and the sending of such a machine to a service computer for execution to provide a desired service, this feature would not have resulted from any obvious combination of the teachings of these references. Accordingly, all claims are believed to patentably distinguish over the applied art.

Claims 18-20 have been added to reflect the monitor lock function described at page 6 of the specification. Neither of the cited references teaches that a service computer can provide the received service container with a monitor lock by which the service container sent by the service server can report back to the service server on the condition of the service computer.

In view of the above, reconsideration and allowance of this application are now believed to be in order, and such actions are hereby solicited. If any points remain in issue which the Examiner feels may be best resolved through a personal or telephone interview, the Examiner is kindly requested to contact the undersigned at the telephone number listed below.

Respectfully submitted,

/DJCushing/

SUGHRUE MION, PLLC
Telephone: (202) 293-7060
Facsimile: (202) 293-7860

WASHINGTON OFFICE
23373
CUSTOMER NUMBER

David J. Cushing
Registration No. 28,703

Date: August 8, 2008